CSEN2061-DBMS-V Semester

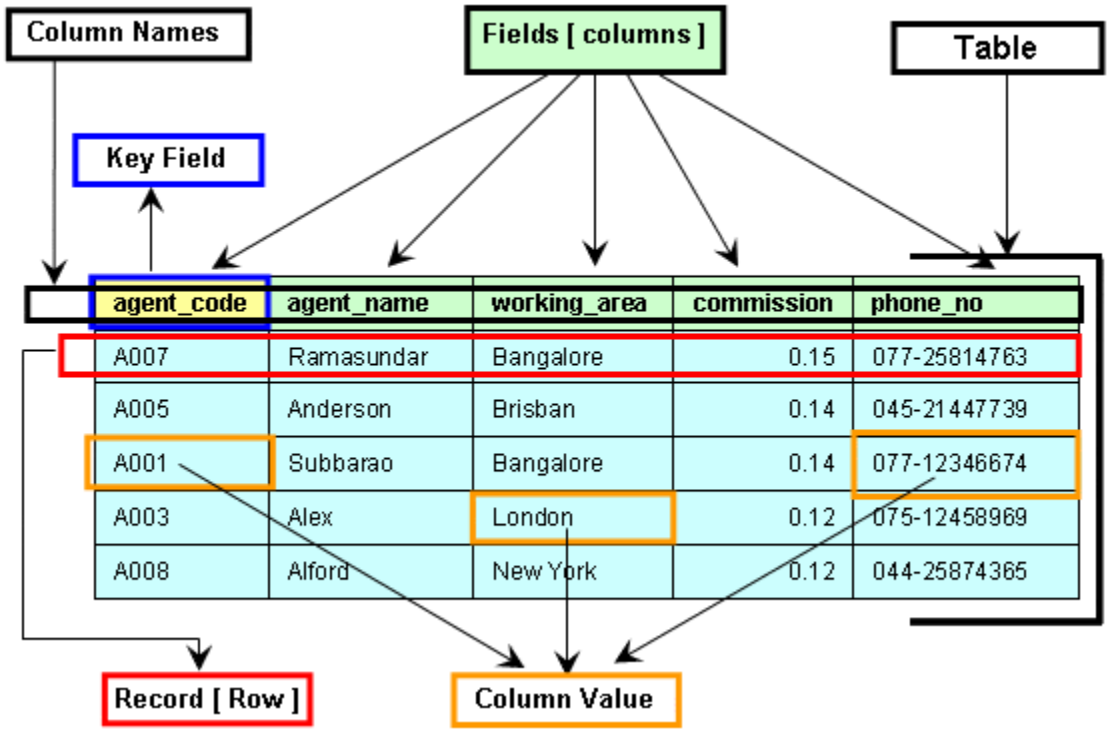
Module 2 Notes

UNIT 2 Relational Model and Basic SQL

Relational model: Integrity constraints over relations and enforcement, querying relation data, logical database design, views, destroying/altering tables and views.

Basic SQL: Introduction to SQL, Basic SQL Queries: **DML, DDL, DCL, TCL**

Relational Model

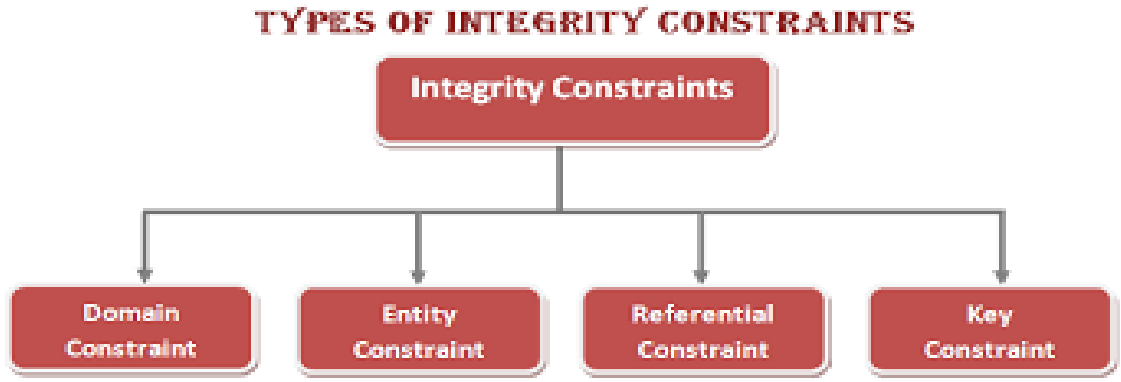


Integrity Constraints

In Database Management Systems, **integrity constraints** are a pre-defined **set of rules that are applied** on the table fields(columns) to ensure that the **overall validity or correctness** of the data present in the database table is maintained.

S_ID	NAME	GENDER	CITY	SEM	MARKS
S0001	Reena	Female	Ahmedabad	3	78
S0002	Vijay	Male	Surat	5	45
S0003	Rahul	Male	Gandhinagar	1	35
S0004	Sanket	Male	Mehsana	3	69
S0005	Chaitali	Female	Rajkot	1	20
S0006	Keyur	Male	Ahmedabad	5	86

S_ID Column values should be unique and not null → Primary Key
NAME Column values should be a collection of letters. Duplicates are allowed.
GENDER Column values should be a collection of letters, and they should be either Male or Female. Duplicates are allowed.
CITY Column values should be a collection of letters. Duplicates are allowed.
SEM Column values should be a single-digit number other than 0. Its values should range between 1 and 8.
MARKS Column values should be a number. Its values should range between 0 to 100.
These observations are to be enforced on the table so that it holds valid or correct data. These enforced rules are called Integrity Constraints.



Types of Integrity Constraints and their implementation in SQL

1. Domain Constraints

Domain constraints refer to the type of values that can be taken on by a specific attribute in a given relation. These constraints help to define and limit the datatype, the acceptable range, as well as the format of a value.

Example:

- Any table that has a column comprising the name 'age' can impose the constraint that it only contains positive integers.

```
create table Person (
  person_id number(3) primary key,
  name varchar2(30),
  age number(3)  check (age > 0),
  gender char(1) check (gender in ('m', ' M', 'f', 'F'))
);
```

Here, we have used the **check** constraints to prevent the age value from being negative, and gender values should only be 'm', 'M', 'f', or 'F'.

Some more examples:

```
marks number(3) check (marks between (0 and 100))
text_book_price number(3) check (price>0 and price<=500)
```

2. Entity Integrity Constraints

Candidate key's integrity, also known as entity integrity, guarantees that each entity (row) within a table of a database is distinguishable. This is done using a primary key that must ensure that all rows in the table have different key values and none of those values can be NULL.

Example:

- A table used for storing records of students in school needs to have one field that is unique for each student.

```
create table student (
  studentID number(5) primary key,
  name varchar2(20),
  E_mail varchar2(15)
);
```

Here, the 'studentID' field is the key field, which means that in this table, no two records of the students are similar.

3. Referential Integrity Constraints

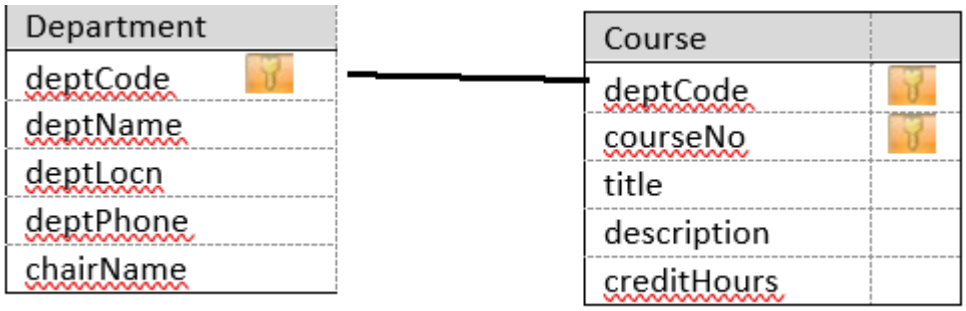
It gives consistent relationships between the related tables so it is called referential integrity. It provides the ownership or definition of values in the foreign key to German counterparts to the primary key. This eliminates improper referencing and keeps up essential information connections.

Example:

- An example of a table known as a **Course** table is a foreign key pointing to a **Department** table.

```
create table Department (
   deptCode varchar2(4) primary key,                    // PARENT TABLE
   deptName varchar2(20),
   deptLocn  varchar2(30),
   deptPhone number(10);
    chairName varchar2(20)
);

create table Course (
   courseNo number(5) primary key,
   title varchar2(20),
   description varchar2(50)
   deptCode varchar2(4)                                  // CHILD TABLE
    foreign key (deptCode) references Department(deptCode)
);
```



4. Key Constraints

Key constraints define that one or more attributes or some sets of attributes must be unique within a table to identify a record. The two main types of keys used are:

- **Primary Key:** A name that is used to identify a table.
- **Unique Key:** This guarantees the uniqueness of a column but permits the attribute to contain a null value.

Example:

```
create table Student (
   Roll_no number(5) primary key,
   Name varchar2(20),
   Class number(1),
    Phone_no number(10) unique
);
```

Example 1: Primary and Unique Key Constraints

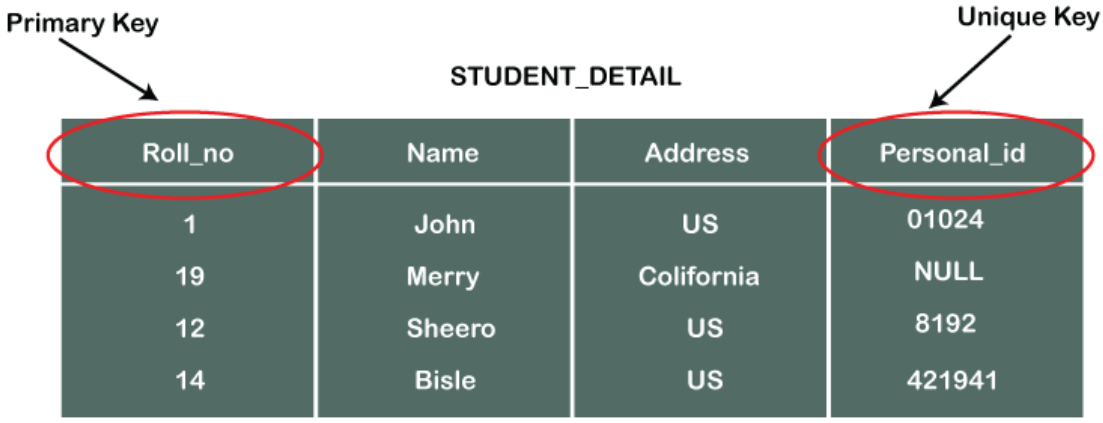
Student

Roll_no	Name	Class	Phone_no
1	Andrew	5	9854672256
2	Andrew	6	9955512456
3	Augusto	5	

Primary Key(No NULL value)

Unique key(Can have NULL value)

Example 2: Primary and Unique Key Constraints



5. NOT NULL Constraints

The NOT NULL constraint guarantees that a column can not contain the NULL value. By its name, it is most often applied to attributes that are required.

Example 1: NOT NULL

```
create table Student (
   Roll_no number(3) primary key,
   Name varchar2(20) not null,
   Class number(1),
   Age number(3)
);
```

Student

Roll_no	Name	Class	Age
1	Andrew	5	12
2	Andrew	5	12
3	Augusto	5	11

Example 2: NOT NULL

Student			
Enroll	S_name	Address	Contact
1001	Rahul	Gwalior	NULL
1002	Rakesh	NULL	9977862211
1003	bharta	gwalior	NULL
1004	atulia	bihar	465748957

Note: Integrity constraints can be enforced after the construction of table(s) using ALTER command.

VIEWS

In a Database Management System (DBMS), a **view is a virtual table** whose content is defined by a query. Unlike a regular table, **a view does not physically store data**; **instead, it derives its data from one or more underlying base tables (or other views) dynamically when it is accessed.**

View from a single table. They are of two types, namely,

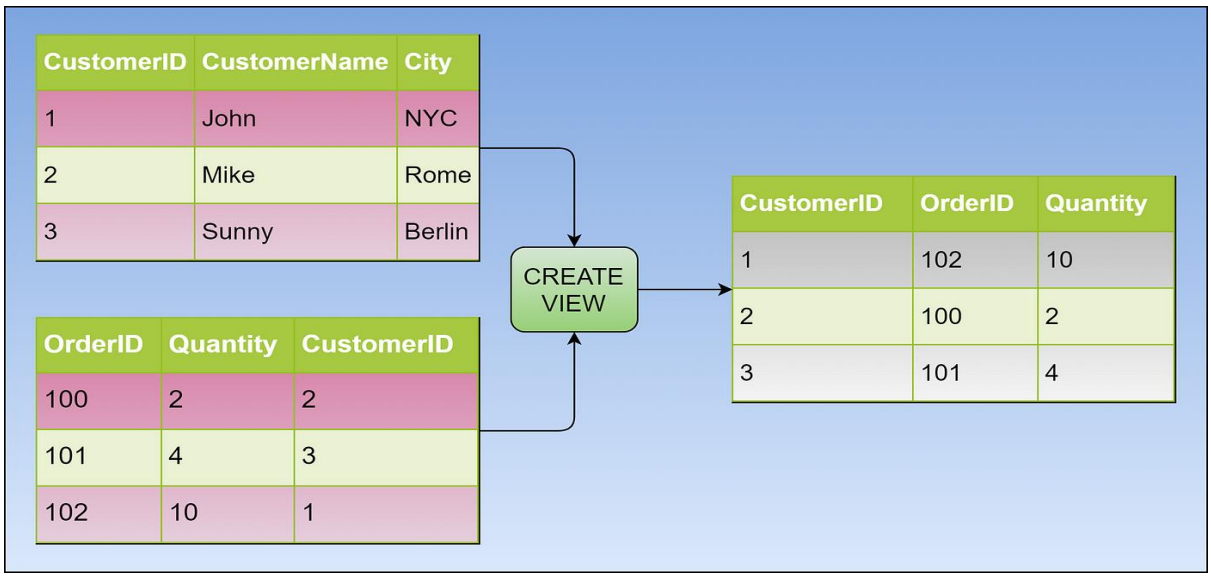
- **View with required fields**
- **View with required records**

Table Name: Employee

Employee_ID	Employee_Name	Job_Title	Salary	Bonus	Age	Manager_ID
1201	Divya	President	50000	NULL	29	NULL
1205	Amyra	Manager	30000	2500	26	1201
1211	Rahul	Analyst	20000	1500	23	1205
1213	Manish	Salesman	15000	NULL	22	1205
1216	Megha	Analyst	22000	1300	25	1201
1217	Mohit	Salesman	16000	NULL	22	1205

View from multiple tables. They are of two types, namely,

- **View with required fields**
- **View with required records**



Implementation of Views in SQL

SQL*Plus: Release 11.2.0.2.0 Production on Tue Jul 29 12:21:41 2025


```
SQL> connect
Enter user-name: system
Enter password:
Connected.

SQL> set linesize 120
SQL> select * from emp1;

     ESSN ENAME            DOB       CITY                 PIN     MOBILE     SALARY
DEPNO
---------- -------------------- --------- -------------------- ---------- ---------- ---------- ----------
      120 Pranav           01-JAN-80 Hyderabad            661201 9877989889    1100000
2
      112 Veena            10-JUL-81 Secunderabad         663301 7977989889
1
      104 Ananya           05-DEC-79 Pune                 763301 8077989889    1900000
1
      105 Prabhu           12-SEP-83 Chennai              563301 8077999889    1600000
3
      100 Surya            10-JAN-24 Bangalore            561201 8.9899E+11    2000000
4
      115 Surya            10-JUN-24 Bangalore            561201 8.9899E+10    2000000
2

6 rows selected.

SQL> select * from department;

DEPTNUMBER DEPTNAME
---------- ----------
         1 HR
         2 SALES
         3 Marketing
         4 Finance
```

- **Create view**

SQL> create view emp1view1 as select essn,ename, city from emp1;

View created.

SQL> select * from emp1view1;

```
     ESSN ENAME                CITY
---------- -------------------- --------------------
       120 Pranav               Hyderabad
       112 Veena                Secunderabad
       104 Ananya               Pune
       105 Prabhu               Chennai
       100 Surya                Bangalore
       115 Surya                Bangalore
```

6 rows selected.

SQL> create view emp1view2 as select * from emp1 where essn in(120,112);

View created.

SQL> select * from emp1view2;

```
     ESSN ENAME                DOB       CITY                 PIN     MOBILE     SALARY
DEPNO
---------- -------------------- --------- -------------------- ---------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad              661201 9877989889    1100000
2
       112 Veena                10-JUL-81 Secunderabad           663301 7977989889
1
```

SQL> create view emp1departmentview1 as select essn,ename,deptname from emp1,department where depno=deptnumber;

View created.

```
SQL> select * from emp1departmentview1;

      ESSN ENAME                DEPTNAME
---------- -------------------- ----------
       120 Pranav               SALES
       112 Veena                HR
       104 Ananya               HR
       105 Prabhu               Marketing
       100 Surya                Finance
       115 Surya                SALES

6 rows selected.

SQL> create view emp1departmentview2 as select * from emp1,department where
depno=deptnumber and depno=1;

View created.

SQL> set linesize 120
SQL> select * from emp1departmentview2;

      ESSN ENAME                DOB       CITY                    PIN     MOBILE     SALARY
DEPNO DEPTNUMBER
---------- -------------------- --------- -------------------- ---------- ---------- ---------- ----------
--
DEPTNAME
----------
       112 Veena                10-JUL-81 Secunderabad           663301 7977989889
1         1
HR
       104 Ananya               05-DEC-79 Pune                  763301 8077989889    1900000
1         1
HR
```

- **Removing view**

```
SQL> drop view emp1departmentview2;
view dropped
SQL> desc emp1departmentview2;
The object does not exist
```

Implementation of Constraints in SQL

Constraints in SQL can be implemented in two ways, namely,

- **Using the create command (See W3SCHOOLS website)**
- **Using the alter command**

Implementing constraints using the alter command

SQL*Plus: Release 11.2.0.2.0 Production on Wed Jul 30 15:28:28 2025
Copyright (c) 1982, 2014, Oracle. All rights reserved.

SQL> connect
Enter user-name: system
Enter password:
Connected.

SQL> set linesize 120

SQL> select * from emp1;

ESSN	ENAME	DOB	CITY	PIN	DEPNO	SALARY
120	Pranav	01-JAN-80	Hyderabad	661201	2	100000
112	Veena	10-JUL-81	Secunderabad	663301	1	120000
104	Ananya	05-DEC-79	Pune	763301	1	200000
105	Prabhu	12-SEP-83	Chennai	563301	3	200000
100	Surya	10-JAN-24	Bangalore	561201	4	120000
115	Silpa	10-JUN-24	Vijayawada	561201	2	120000

6 rows selected.

- **Primary key constraint**

SQL> alter table emp1 add constraint pk_essn primary key(essn);

Table altered

Command to see constraints on a particular table

SQL> select constraint_name,constraint_type from user_constraints where table_name='EMP1';

CONSTRAINT_NAME	C
PK_ESSN	P

Primary Key Constraint Violation

Duplicates not allowed

```
SQL> insert into emp1 values(120,'yyy','01-jan-1992','hyd',111111,1,100000);
insert into emp1 values(120,'yyy','01-jan-1992','hyd',111111,1,100000)
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.PK_ESSN) violated
```

Null value not allowed

```
SQL> insert into emp1 values('','yyy','01-jan-1992','hyd',111111,1,100000);
insert into emp1 values('','yyy','01-jan-1992','hyd',111111,1,100000)
                        *
ERROR at line 1:
ORA-01400: cannot insert NULL into ("SYSTEM"."EMP1"."ESSN")
```

- **Check constraint**

```
SQL> alter table emp1 add constraint ck_salary check(salary between 100000 and 250000);

Table altered.
```

Command to see constraints on a particular table

```
SQL> select constraint_name,constraint_type from user_constraints where table_name='EMP1';
CONSTRAINT_NAME              C
------------------------------ -
PK_ESSN                      P
CK_SALARY                    C
```

Check constraint violation

```
SQL> insert into emp1 values(200,'yyy','01-jan-1992','hyd',111111,1,260000);
insert into emp1 values(200,'yyy','01-jan-1992','hyd',111111,1,260000)
*
ERROR at line 1:
ORA-02290: check constraint (SYSTEM.CK_SALARY) violated
```

Consider the following table once again. Let us see NOT NULL and UNIQUE constraints now.

```
SQL> set linesize 120

SQL> select * from emp1;

      ESSN ENAME                DOB       CITY                      PIN      DEPNO     SALARY
---------- -------------------- --------- -------------------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad                661201          2     100000
```

112	Veena	10-JUL-81	Secunderabad	663301	1 120000
104	Ananya	05-DEC-79	Pune	763301	1 200000
105	Prabhu	12-SEP-83	Chennai	563301	3 200000
100	Surya	10-JAN-24	Bangalore	561201	4 120000
115	Silpa	10-JUN-24	Vijayawada	561222	2 120000

6 rows selected.

- **Not Null Constraint**

SQL> **alter table emp1 modify ename varchar2(20) not null;**

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where table_name='EMP1';

CONSTRAINT_NAME	C
SYS_C007855	C

- **Unique Constraint**

SQL> alter table emp1 add constraint uk_pin unique(pin);

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where table_name='EMP1';

CONSTRAINT_NAME	C
SYS_C007855	C
UK_PIN	U

Duplicates not allowed

SQL> insert into emp1 values(200,'yyy','01-jan-1992','hyd',561222,1,260000);
insert into emp1 values(200,'yyy','01-jan-1992','hyd',561222,1,260000)
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.UK_PIN) violated

Any number of null values is allowed

```
SQL> insert into emp1 values(200,'yyy','01-jan-1992','hyd',,1,260000);

1 row created.

SQL> insert into emp1 values(201,'yyy','01-jan-1992','hyd',,1,260000);

1 row created.

SQL> select * from emp1;

      ESSN ENAME                DOB       CITY                      PIN      DEPNO     SALARY
---------- -------------------- --------- -------------------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad            661201          2     100000
       112 Veena                10-JUL-81 Secunderabad         663301          1     120000
       104 Ananya               05-DEC-79 Pune                 763301          1     200000
       105 Prabhu               12-SEP-83 Chennai              563301          3     200000
       100 Surya                10-JAN-24 Bangalore            561201          4     120000
       115 Silpa                10-JUN-24 Vijayawada           561222          2     120000
       200 yyy                  01-JAN-92 hyd                                  1     260000
       201 yyy                  01-JAN-92 hyd                                  1     260000

8 rows selected.
```

Primary Key Vs Unique Constraints

- Neither allows duplicates
- Primary key further does not allow null values. Whereas a unique key allows any number of null values.
- Primary Key = Unique Key + Not Null

Removing Not Null and Unique constraints

```
SQL> select constraint_name, constraint_type from user_constraints where
table_name='EMP1';

CONSTRAINT_NAME              C
------------------------------ -
SYS_C007855                  C
UK_PIN                       U
```

Removing Not Null Constraint

```
SQL> alter table emp1 drop constraint SYS_C007855;

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where
table_name='EMP1';
```

```
CONSTRAINT_NAME              C
------------------------------ -
UK_PIN                         U
```

Removing Unique Constraint

```
SQL> alter table emp1 drop constraint uk_pin;

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where
table_name='EMP1';

no rows selected
```

- **Foreign Key Constraint (Requires two tables)**

Consider the following two tables

SQL> select * from emp1;

ESSN	ENAME	DOB	CITY	PIN	DEPNO	SALARY
120	Pranav	01-JAN-80	Hyderabad	661201	2	100000
112	Veena	10-JUL-81	Secunderabad	663301	1	120000
104	Ananya	05-DEC-79	Pune	763301	1	200000
105	Prabhu	12-SEP-83	Chennai	563301	3	200000
100	Surya	10-JAN-24	Bangalore	561201	4	120000
115	Silpa	10-JUN-24	Vijayawada	561222	2	120000

6 rows selected.

SQL> select * from department;

DEPTNUMBER	DEPTNAME
1	HR
2	SALES
3	Marketing
4	Finance

SQL> alter table emp1 add constraint **fk_emp1_depno foreign key(depno) references department(deptnumber);**

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where table_name='EMP1';

```
CONSTRAINT_NAME              C
------------------------------ -
FK_EMP1_DEPNO                R
```

Note the following points

- 'emp1' table is a **child table.**
- 'department' table is a **parent table.**
- Both tables have a common field, namely, the 'depno' of emp1 and the 'deptnumber' of the department.
- 'depno' of emp1(child table) is a foreign key field whereas 'deptnumber' of department(parent table) is a primary key
- 'deptno' field of emp1 should draw only those values that are in the 'deptnumber' field of department.

Violations of Foreign Key Constraint

```
SQL> insert into emp1 values(200,'yyy','01-jan-1992','hyd',666666,8,260000);
insert into emp1 values(200,'yyy','01-jan-1992','hyd',666666,8,260000)
*
ERROR at line 1:
ORA-02291: integrity constraint (SYSTEM.FK_EMP1_DEPNO) violated - parent key not found
```

Note the following points of FOREIGN KEY CONSTRAINT

In a foreign key, null value(s) are allowed

SQL> insert into emp1 values(200,'yyy','01-jan-1992','hyd',666666,'',260000);

1 row created.

SQL> select * from emp1;

ESSN	ENAME	DOB	CITY	PIN	DEPNO	SALARY
120	Pranav	01-JAN-80	Hyderabad	661201	2	100000
112	Veena	10-JUL-81	Secunderabad	663301	1	120000
104	Ananya	05-DEC-79	Pune	763301	1	200000
105	Prabhu	12-SEP-83	Chennai	563301	3	200000
100	Surya	10-JAN-24	Bangalore	561201	4	120000

```
115 Silpa          10-JUN-24 Vijayawada          561222        2    120000
200 yyy            01-JAN-92 hyd                 666666                  260000
```

7 rows selected.

In a foreign key, duplicate value(s) are allowed

SQL> insert into emp1 values(201,'yyy','01-jan-1992','hyd','',1,260000);

1 row created.

SQL> select * from emp1;

```
      ESSN ENAME              DOB       CITY                    PIN      DEPNO     SALARY
---------- -------------------- --------- -------------------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad              661201          2     100000
       112 Veena                10-JUL-81 Secunderabad           663301          1     120000
       104 Ananya               05-DEC-79 Pune                   763301          1     200000
       105 Prabhu               12-SEP-83 Chennai                563301          3     200000
       100 Surya                10-JAN-24 Bangalore              561201          4     120000
       115 Silpa                10-JUN-24 Vijayawada             561222          2     120000
       200 yyy                  01-JAN-92 hyd                    666666                260000
       201 yyy                  01-JAN-92 hyd                                    1     260000
```

8 rows selected.

Removing Foreign Key Constraint

SQL> select constraint_name, constraint_type from user_constraints where table_name='EMP1';

```
CONSTRAINT_NAME              C
------------------------------ -
FK_EMP1_DEPNO                  R
```

SQL> alter table emp1 drop constraint fk_emp1_depno;

Table altered.

SQL> select constraint_name, constraint_type from user_constraints where table_name='EMP1';

no rows selected

Basic SQL Queries: **DDL, DML, DCL, TCL**

DDL- Data Definition Language Commands

- ## Create

create table table_name (field_1 data_type(size), field_2 data_type(size), .. .);

- ## Desc

desc table_name;

- ## Rename

rename old_table_name to new_table_name;

- ## Alter

It is used to change the structure of the table, namely,

1. Add new fields/columns to the table

alter table table_name **add** new_field_name data_type(size);

2. Remove existing column of the table

alter table table_name **drop column** field_name;

3. Change the size of data type of the column of the table

alter table table_name **modify** field data_type**(new_size);**

4. Rename the column name from one to another

alter table table_name **rename column** old_field_name **to** new_field_name;

DML-Data Manipulation Language Commands

- ## insert

  insert into table_name **values(field1_value,field2_value, …….fieldn_value);**

Note:

Number values are given, like, 10, 100, 123.20, 10.25 and so on
String values are given, like, ‘Ram’, ‘Ramesh’, ‘Rakesh123@gmail.com’ and so on
Date value is given, like, ’10-jan-1992’, ’25-oct-1996’ and so on

- ## update

  update table_name **set** field2_name=value **where field1_name=value;**

- ## select

It is used to **retrieve or access records of the table.**

- **select * from** table_name;

all fields and all rows of the table are displayed

- **select field1, field2 from** table_name;
  field1, field2 for all the rows of the table are displayed

- **select * from** table_name **where condition;**
  all fields of single or a set of rows of the table are displayed for which condition holds

- **select field1, field2 from** table_name **where condition;**
  field1, field2 of a single or a set of rows of the table are displayed for which condition holds

- **delete**

To delete single or a set of records from the table

delete from table_name **where condition;**

To delete all records from the table

delete from table_name**;**

DDL- Data Definition Language Commands - Continuation

- **Truncate**

truncate table table_name; //Content of the table gets removed, but structure remains

- **Drop**

drop table table_name; // both content and structure of the table get removed.

DCL - Data Control Language of SQL

DCL (Data Control Language) includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions and other controls of the database system. These commands are used to control access to data in the database by granting or revoking permissions.
Common DCL Commands

Command	Description	Syntax
GRANT	Assigns new privileges to a user account, allowing access to specific database objects, actions, or functions.	GRANT privilege_type [(column_list)] ON [object_type] object_name TO user [WITH GRANT OPTION];

Command	Description	Syntax
REVOKE	Removes previously granted privileges from a user account, taking away their access to certain database objects or actions.	REVOKE [GRANT OPTION FOR] privilege_type [(column_list)] ON [object_type] object_name FROM user [CASCADE];

Example of DCL
GRANT SELECT, UPDATE ON employees TO user_name;

TCL (Transaction Control Language Commands) of SQL

Transactions group a set of tasks into a single execution unit. Each transaction begins with a specific task and ends when all the tasks in the group are successfully completed. If any of the tasks fail, the transaction fails. Therefore, a transaction has only two results: success or failure.

COMMIT	Saves all changes made during the transaction	COMMIT;
ROLLBACK	Undoes all changes made during the transaction	ROLLBACK;
SAVEPOINT	Creates a savepoint within the current transaction	SAVEPOINT savepoint_name;

Implementation of TCL in SQL

SQL*Plus: Release 11.2.0.2.0 Production on Sun Aug 3 22:45:10 2025

SQL> connect
Enter user-name: system
Enter password:
Connected.

```
SQL> set linesize 120
SQL> select * from emp1;

     ESSN ENAME              DOB       CITY                    PIN      DEPNO     SALARY
---------- -------------------- --------- -------------------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad            661201          2     100000
       112 Veena                10-JUL-81 Secunderabad         663301          1     120000
       104 Ananya               05-DEC-79 Pune                 763301          1     200000
       105 Prabhu               12-SEP-83 Chennai              563301          3     200000
       100 Surya                10-JAN-24 Bangalore            561201          4     120000
       115 Silpa                10-JUN-24 Vijayawada           561222          2     120000
       200 yyy                  01-JAN-92 hyd                  666666                260000
       201 yyy                  01-JAN-92 hyd                                  1     260000

8 rows selected.

SQL> savepoint s1;

Savepoint created.

SQL> delete from emp1 where essn=201;

1 row deleted.


SQL> savepoint s2;

Savepoint created.

SQL> delete from emp1 where essn=200;

1 row deleted.

SQL> select * from emp1;

     ESSN ENAME              DOB       CITY                    PIN      DEPNO     SALARY
---------- -------------------- --------- -------------------- ---------- ---------- ----------
       120 Pranav               01-JAN-80 Hyderabad            661201          2     100000
       112 Veena                10-JUL-81 Secunderabad         663301          1     120000
       104 Ananya               05-DEC-79 Pune                 763301          1     200000
       105 Prabhu               12-SEP-83 Chennai              563301          3     200000
       100 Surya                10-JAN-24 Bangalore            561201          4     120000
       115 Silpa                10-JUN-24 Vijayawada           561222          2     120000

6 rows selected.
```

SQL> rollback to s2;

Rollback complete.

SQL> select * from emp1;

ESSN	ENAME	DOB	CITY	PIN	DEPNO	SALARY
120	Pranav	01-JAN-80	Hyderabad	661201	2	100000
112	Veena	10-JUL-81	Secunderabad	663301	1	120000
104	Ananya	05-DEC-79	Pune	763301	1	200000
105	Prabhu	12-SEP-83	Chennai	563301	3	200000
100	Surya	10-JAN-24	Bangalore	561201	4	120000
115	Silpa	10-JUN-24	Vijayawada	561222	2	120000
200	yyy	01-JAN-92	hyd	666666		260000

7 rows selected.

SQL> rollback to s1;

Rollback complete.

SQL> select * from emp1;

ESSN	ENAME	DOB	CITY	PIN	DEPNO	SALARY
120	Pranav	01-JAN-80	Hyderabad	661201	2	100000
112	Veena	10-JUL-81	Secunderabad	663301	1	120000
104	Ananya	05-DEC-79	Pune	763301	1	200000
105	Prabhu	12-SEP-83	Chennai	563301	3	200000
100	Surya	10-JAN-24	Bangalore	561201	4	120000
115	Silpa	10-JUN-24	Vijayawada	561222	2	120000
200	yyy	01-JAN-92	hyd	666666		260000
201	yyy	01-JAN-92	hyd		1	260000

8 rows selected.

SQL> commit;

Commit completed.